Journal of Information Hiding and Multimedia Signal Processing ©2010 ISSN 2073-4212
Ubiquitous International Volume **1**, Number **3**, July **2010**

Image Compression Based on Mean Value Predictive Vector Quantization

Zhe-Ming Lu

School of Aeronautics and Astronautics
Zhejiang University
Hangzhou 310027, P. R. China.
zheminglu@zju.edu.cn

Yue-Nan Li

Department of Electronic and Information Engineering
Harbin Institute of Technology Shenzhen Graduate School
Shenzhen 518055, P. R. China.
liyuenan@yahoo.com

Received May 2009; revised April 2010

Abstract. *The main drawback of traditional predictive vector quantization (PVQ) is the time consuming encoding process. In order to reduce the computational complexity, an algorithm called mean value predictive vector quantization (MVPVQ) is proposed in this paper. Input vectors are classified into smooth vectors and non-smooth vectors before encoding. Different kinds of input vectors are encoded with different encoding schemes. Simulations demonstrate that the proposed method can achieve satisfying image quality at low bit rates.*
Keywords: Image compression, Predictive vector quantization, Mean value.

1. **Introduction.** VQ has been considered as an efficient block-based lossy compression technique. With the advantages of simple coding process and high compression ratio, VQ is widely used in the fields of image compression and speech coding, and it can always archive better performance than scalar quantization. In VQ, a vector is compared with codewords in a codebook to find the best matching codeword, and compression is achieved by transmitting or storing the index of that codeword rather than the codeword itself. VQ can be catalogued into memoryless VQ and memory VQ. Memoryless VQ quantizes each input vector independently, whereas memory VQ exploits the relationships between neighboring vectors [1].

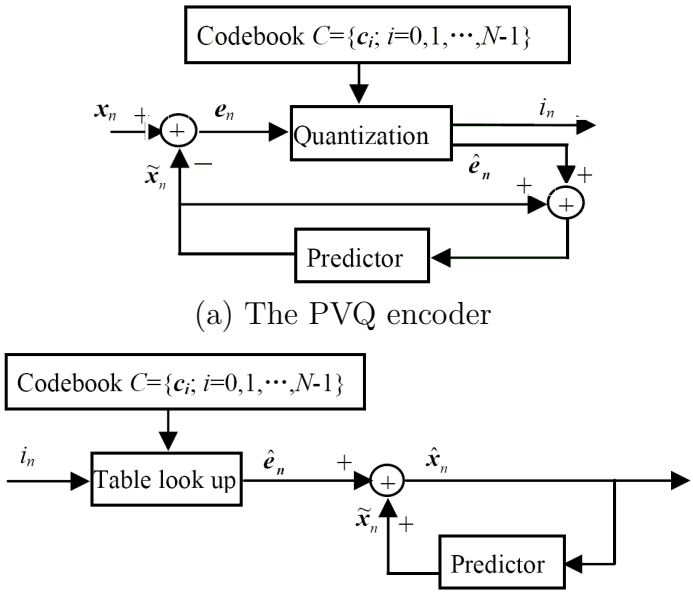
PVQ [2, 3] is a kind of commonly used memory VQ. The most evident drawback of PVQ is the high computational complexity which limits PVQ being used in real time image compression systems. Shorter coding time can be archived by reducing the number of codewords, but the reconstructed image quality will degrade in the case of fewer codewords. Another way of reducing the coding time is to use fast codeword searching schemes. The fact that the mean value of a residual vector is close to zero makes many fast codeword search schemes that based on mean value no longer compatible with PVQ, hence it is necessary to propose fast coding algorithms for PVQ. MVPVQ proposed in this letter can reduce the coding time and bit rate.

The rest parts of this letter are organized as follows. Section 2 chiefly reviews the theory of PVQ. The proposed MVPVQ algorithm is described in Section 3. Section 4 shows simulation results on still image. Finally, conclusions are given in Section 5.

2. **Predictive Vector Quantization.** PVQ is the extension of differential pulse code modulation (DPCM) in VQ. In PVQ, input vectors are first predicted by previously reconstructed vectors, and then the encoder quantizes the prediction error between the input vector and its prediction value. It can be proved that the reconstruction error of PVQ is equal to the quantization error of the encoder, namely

$$\boldsymbol{q}_n = \hat{\boldsymbol{x}}_n - \boldsymbol{x}_n = \tilde{\boldsymbol{x}}_n + \hat{\boldsymbol{e}}_n - (\tilde{\boldsymbol{x}}_n + \boldsymbol{e}_n) = \hat{\boldsymbol{e}}_n - \boldsymbol{e}_n \tag{1}$$

Because neighboring pixels tend to be highly correlated, the reconstruction error of PVQ is less than that of memoryless VQ. The structures of PVQ encoder and decoder are shown in Fig.1.



(a) The PVQ encoder

(b) The PVQ decoder

FIGURE 1. Structures of PVQ encoder and decoder

PVQ has been widely used in image compression and 3D mesh model compression. Reference [4] proposed an efficient hybrid image vector quantization (VQ) technique based on a classification in the DCT domain. This algorithm combines two kinds of VQ, predictive VQ (PVQ) and discrete cosine transform domain VQ (DCTVQ), and adopts a simple classifier which employs only three DCT coefficients in the 8×8 block. For each image block, the classifier switches to the PVQ coder if the block is relatively complex, and otherwise switches to the DCTVQ coder. Reference [5] proposed a new fast approach to the nearest codevector search for 3D mesh compression using an orthonormal transformed codebook, where 3D meshes are vector quantized based on the parallelogram prediction. Based on the same prediction idea as [5], Reference [6] proposed a dynamically restricted codebook based vector quantization scheme for mesh geometry compression.

3. **Mean Value Predictive Vector Quantization.** By analyzing an actual image we can draw a conclusion that most parts of the image are smooth areas which vary little. One typical example is the background of an image. Non-smooth areas only take a small portion, whereas the human perceptual system is sensitive to not smooth areas but non-smooth areas. The proposed MVPVQ classifies input vectors into smooth vectors and non-smooth vectors before encoding. In order to reduce the computational complexity,smooth vectors will not be encoded in the way of PVQ. We also classify training vectors in the process of codebook generation to improve the codebook performance.

We adopt the vector variance σ^2 to judge whether an input vector is smooth or not and it can be calculated as

$$\sigma^2 = \frac{1}{k}\sum_{i=1}^{k}(x_i - \mu)^2 \tag{2}$$

where k is the vector dimension, μ is the mean value of the input vector, which can be calculated as follows.

$$\mu = \frac{1}{k}\sum_{i=1}^{k} x_i \tag{3}$$

If $\sigma^2 < TH_1$ is satisfied, the input vector can be classified as a smooth vector, where TH_1 is the threshold defined before quantizing. It implies that each component of the input vector varies little from the mean value when the input vector is classified as a smooth vector. Each component of the input vector can be presented by p which is close to the mean value of the input vector in this case. Based on the fact that neighboring pixels are highly correlated in an image, the prediction of mean value can be calculated as follows.

$$p = \frac{1}{9}\sum_{i=0}^{8} X_i \tag{4}$$

where X_i is the neighboring pixels of the input vector and the relationship between X_i and the input vector is shown in Fig.2

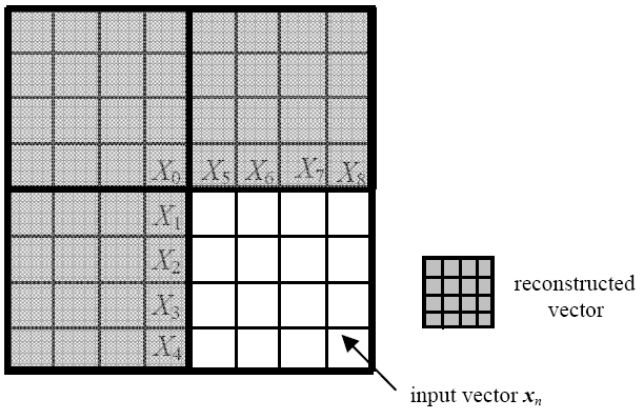


FIGURE 2. Input vector and its neighboring pixels

Based on the above description, the encoding steps of MVPVQ can be depicted as follows:

Step 1: Quantize the vectors that located in Row 1 and Column 1 (i.e., main blocks) by the traditional PVQ encoder.

Step 2: Predict the mean value of the input vector by the upper and left vectors according to Eq.(4). If the predicted mean value is close to the actual mean value, namely

$$|p - \mu| < TH_2 \tag{5}$$

go to Step3; Otherwise, go to Step4.

Step 3: Calculate the variance σ^2 of the input vector. If $\sigma^2 < TH_1$, then set the information bit to be '0' and transmit or store one bit index (information bit), go to Step 5.

Step 4: Quantize the input vector with the traditional PVQ. Set the information bit as '1', and then transmit the information bit and the index of the residual codeword.

Step 5: Go to Step 2 for next input vector until all input vectors have been processed.

The corresponding decoding steps of MVPVQ can be illustrated as follows:

Step 1: Decode the vectors that located in Row 1 and Column 1 (main blocks) by the traditional PVQ.

Step 2: Judge the information bit of the received index. If the information bit is '0', go to Step 3; otherwise, go to Step 4.

Step 3: Predict the mean value of the input vector using its neighboring reconstructed blocks and set each component of the reconstructed vector to be p, go to Step 5.

Step 4: Decode the input vector by the traditional PVQ decoder.

Step 5: Go to Step 2 for next input vector until all input vectors have been processed.

Considering the correlations among neighboring pixels, the input vector can be predicted by neighboring pixels. Reference [7] has proposed several prediction schemes based on neighboring pixels. It has been shown that predicting the current vector by its neighboring pixels can improve the prediction accuracy, and the computational complexity can be also kept at a low level. In this paper, the multiple-pixels-distance-weighted boundary PVQ (MPDWBPVQ) that proposed in [7] is adopted to predict input vectors. The prediction schemes are as follows.

$$\tilde{x}(i,j) = \frac{i\tilde{Y}_e + j\tilde{Y}_u}{i+j} \quad (1 < i < 4, 1 < j < 4) \tag{6}$$

$$\tilde{Y}_e = 0.1X_0 + 0.2X_1 + 0.4X_2 + 0.2X_3 + 0.1X_4 \tag{7}$$

$$\tilde{Y}_u = 0.1X_0 + 0.2X_5 + 0.4X_6 + 0.2X_7 + 0.1X_8 \tag{8}$$

where $\tilde{x}(i,j)$ is the predicted pixel value at the coordinate (i,j). Unlike the original MPDWBPVQ method, the proposed method stores the left upper corner vector in the codebook, and quantizes other input vectors in Row 1 and Column 1 by the traditional PVQ. The corresponding prediction schemes can be described as follows.

For vectors in Row 1:

$$\tilde{x}(i,j) = 0.1X_0 + 0.2X_1 + 0.4X_2 + 0.2X_3 + 0.1X_4 \tag{9}$$

For vectors in Column 1:

$$\tilde{x}(i,j) = 0.1X_0 + 0.2X_5 + 0.4X_6 + 0.2X_7 + 0.1X_8 \tag{10}$$

From the above encoding process, we can find that it is not necessary to quantize smooth vectors by the traditional PVQ encoder, so MVPVQ can significantly reduce the computational complexity. Furthermore, if the mean value of the smooth input vector can be accurately predicted, only one bit index has to be transmitted or stored for that input vector. Smooth vectors widely exist in the image background, so the bit rate can be efficiently reduced. In order to get higher compression ratio, the codeword indices are also encoded using the Huffman coding technique.

Codebook generation is essential to PVQ algorithms as reconstructed residual vectors are selected from this codebook. In MVPVQ, smooth vectors whose mean values can be accurately predicted will not be quantized by the PVQ encoder, so only non-smooth vectors is selected to be training vectors. If each training vector is chosen from the image to be compressed, we also judge whether its mean value can be accurately predicted by its neighboring pixels. In this way, we only need to generate the codewords for non-smooth vectors, so the computational complexity will be kept at a low level. Simulations show that the codebook designed in this way (MVPVQ codebook) is also compatible with traditional PVQ algorithms that adopt the same prediction scheme and the reconstructed image quality will be improved.

4. **Simulation Results.** In order to evaluate the performance of MVPVQ, several simulation experiments have been done on a Pentium IV computer whose CPU frequency is 1.8GHz. We select 512×512 monochrome Peppers image (inside the training set) and Lena image (outside the training set) with 256 gray levels as test images. The original Peppers image is divided into 4×4 blocks to form input vectors. The codebook contains 511 residual codewords and 1 codeword equal to the left upper corner vector. Non-smooth vectors or smooth vectors whose mean values can not be accurately predicted are selected to be training vectors. We set thresholds $TH_1 = 320$ and $TH_2 = 5$ to select training vectors and the GLA algorithm [1] is used to cluster residual vectors. Full search algorithm is used in searching the best matching residual codeword. TH_1 and TH_2 are set to be zero to test the performance of the MVPVQ codebook used in the traditional PVQ algorithm with the same prediction scheme. Table 1 and Table 2 compare our MVPVQ method with several commonly used algorithms, including Boundary PVQ(BPVQ) [8], Multi-stage VQ(MSVQ)[9], Multi-stage PVQ (MSPVQ)[10] and MPDWBPVQ [7], in terms of PSNR, coding time and bit rate. The performance of the international still image compression standard JPEG is also compared with that of MVPVQ at low bit rates and the comparison result is shown in Fig.3.

From the simulation results, we can see that our MVPVQ method is a variable bit rate PVQ algorithm and the coding performance varies with TH_1 and TH_2. Selecting suitable thresholds according to different images is essential to achieve higher coding performance. From Table 1, we can see that the proposed MVPVQ method can get higher reconstructed image quality when the bit rate is 0.16bpp lower than that of VQ and the computational complexity is also efficiently reduced. For MSVQ and MSPVQ methods, the unique encoder architectures provide higher encoding speed, but the reconstructed image quality is much lower than other algorithms. From Table 2, we can see that the proposed MVPVQ scheme degrades when used in encoding the image outside the training set. The Lena image has relative abundant non-smooth blocks, so TH_1 is smaller than that used in quantizing the Peppers image.

5. **Conclusions.** A low computational complexity PVQ algorithm, MVPVQ, is proposed in this paper. From the simulation results, we can see that our MVPVQ method is a variable bit rate PVQ algorithm and its coding performance is higher than other VQ methods.

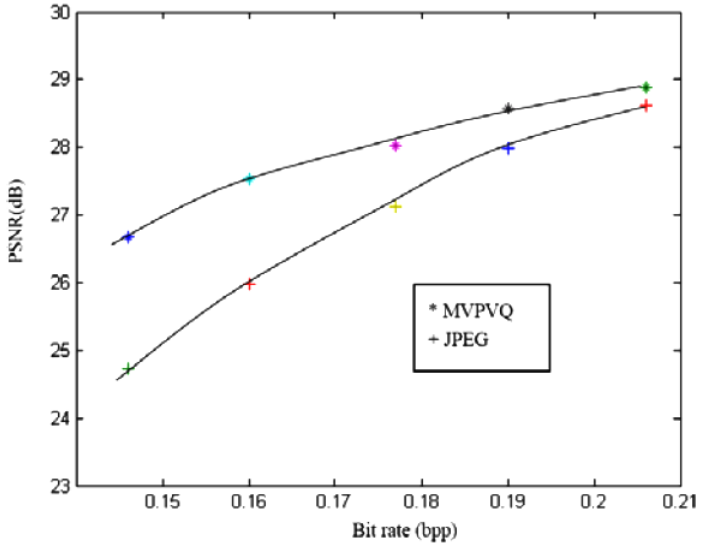


FIGURE 3. Performance comparison of JPEG and MVPVQ

TABLE 1. Performance comparison of several algorithms (Peppers:inside the training set)

Algorithm		PSNR (dB)	Coding time (s)	Bit rate (bpp)
MSVQ		26.442	0.72	0.562
MSPVQ		29.774	1.40	0.562
BPVQ		29.415	2.52	0.562
MPDWBPVQ		32.032	2.64	0.558
MPDWBPVQ	$(Th_1 = 20, TH_2 = 3)$	31.975	1.89	0.392
	$(Th_1 = 15, TH_2 = 2)$	32.408	2.20	0.450
	$(Th_1 = 10, TH_2 = 1)$	32.527	2.54	0.506
	$(Th_1 = 0, TH_2 = 0)$	32.576	2.68	0.523

TABLE 2. Performance comparison of several algorithms (Lena:outside the training set)

Algorithm		PSNR (dB)	Coding time (s)	Bit rate (bpp)
MSVQ		21.714	0.41	0.562
MSPVQ		28.523	2.11	0.562
BPVQ		29.415	2.52	0.562
MPDWBPVQ		29.662	2.76	0.558
MPDWBPVQ	$(Th_1 = 20, TH_2 = 3)$	29.404	2.73	0.492
	$(Th_1 = 15, TH_2 = 2)$	28.449	2.50	0.453
	$(Th_1 = 10, TH_2 = 1)$	28.331	2.56	0.465
	$(Th_1 = 0, TH_2 = 0)$	30.315	2.96	0.521

Experimental results also show that the MVPVQ codebook design algorithm outperforms the traditional open-loop PVQ codebook design algorithm. The reconstructed image quality of MVPVQ is also higher than that of JPEG at low bit rates.

REFERENCES

[1] S. H. Sun and Z. M. Lu, *Vector quantization fundamentals and applications*, Chinese Science Press, Beijing, 2002.

[2] V. Cuperman and A. Gersho, Adaptive differential vector coding of speech, *Conference Record GlobeCom 82*, pp. 1092–1096, 1982.

[3] T. R. Fischer and D. J. Tinnen, Quantized control with Differential Pulse Code Modulation, *Proc. of the 21th Conference on Decision and Control*, pp. 1222–1227, 1982.

[4] Z. M. Lu and H. Pei, Hybrid Image Compression Scheme Based on PVQ and DCTVQ, *IEICE Trans. Information and Systems*, vol. E88-D, no. 10, pp. 2422–2426, 2005.

[5] Z. Li and Z. M. Lu, Fast codevector search scheme for 3D mesh model vector quantization, IET Electronics Letters, vol. 44, no. 2, pp. 104–105, 2008.

[6] Z. M. Lu and Z. Li, Dynamically restricted codebook based vector quantisation scheme for mesh geometry compression" Signal, *Proc. of Image and Video Processing*, vol. 2, no. 3, pp. 251–260, 2008.

[7] B. Yang, Z. M. Lu, D. G. Xu, and S. H. Sun, Neighboring pixels based low complexity predictive vector quantization algorithms for image coding, *ACTA ELECTRONICA SINICA*, vol. 31, no. 5, pp. 707–710, 2003.

[8] A. Gersho and R. M. Gray, Vector Quantization and Signal Compression, *Kluwer Academic Publisher*, Boston, 1992.

[9] B. Mahesh and W. A. Pearlman, Variable-rate tree structured vector quantizer, *IEEE Trans. Information Theory*, vol. 41, no. 4, pp. 917–930, 1995.

[10] S. A. Rizvi and N. M. Nasrabadi, Predictive residual vector quantization, *IEEE Trans. Image Processing*, vol. 4, no. 11, pp. 1482–1495, 1995.